

User stories collection via interactive chatbot to support requirements gathering

Ferliana Dwitama, Andre Rusli

Department of Informatics, Universitas Multimedia Nusantara, Indonesia

Article Info

Article history:

Received Jul 7, 2019

Revised Jan 7, 2020

Accepted Feb 20, 2020

Keywords:

AIML

Chatbot

Nazief & Adriani algorithm

Requirements elicitation

User stories

ABSTRACT

Nowadays, software products have become an essential part of human life. To build software, developers must have a good understanding of the requirements of the software. However, software developers tend to jumpstart system construction without having a clear and detailed understanding of the requirements. The user story concept is one of the practices of the requirements elicitation. This paper aims to present the work conducted to develop an Android chatbot application to support the requirements elicitation activity in software engineering, making the work less time-consuming and structured even for users not accustomed to requirements engineering. The chatbot uses Nazief & Adriani stemming algorithm to pre-process the natural language it receives from the users and artificial mark-up language (AIML) as the knowledge base to process the bot's responses. A preliminary acceptance test based on the technology acceptance model results in an 83.03% score for users' behavioral intention to use.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Ferliana Dwitama,

Department of Informatics,

Universitas Multimedia Nusantara,

Scientia Boulevard St., Gading, Serpong, Tangerang, Banten 15227, Indonesia.

Email: ferliana.dwitama@student.umn.ac.id

1. INTRODUCTION

Software is an important aspect that has an impact on human life, such as work, daily activity, finance, and other essential fields. There are some steps to build software, and developers must have a clear understanding of the requirements of the software. However, the developer tends to build software without having a clear understanding of the requirements in detail because for them having understood requirements in detail is a waste of time, in which requirements are constantly changing [1]. These things lead to software failure because when the developer does not have a clear understanding of the system, an error can occur in system design and the next steps [2].

Requirements engineering is an activity that has the aim to know, understand, analyze, and documenting what are the requirements that needed by stakeholder [1], so involvement of stakeholders is crucial in software project especially in requirements engineering process [1, 3]. One of the steps requirements engineering is requirements elicitation which user story is one of its practice. User stories introduced the first time to extreme programming in agile development methods and started used in other methods, such as Scrum, [4]. User stories focus on interaction with the user, so users can participate and describe the functions that can be useful for stakeholders. A chatbot is a computer agent that can interact with the user and make human-bot conversation feels like they were a human-human conversation. Based on

previous research, natural language processing used for chatbot can interact and understand like a human [5–7]. Based on a survey by Oracle, more than 70% of the respondents said that people or businesspeople already used chatbot or have a plan to use a chatbot in 2020 [8]. When using a chatbot, the customer does not need to waste their time to meet face to face to get specific service which makes users save the cost; these things make chatbot has brought positive impact for both customers and the company [9].

Sending messages is the core of the mobile experience [10], and the increase in internet and mobile devices in the present makes the interaction between humans easier [11]. Based on a survey in 2017, mobile device users in Indonesia already reach 341,4 million [12]. Artificial mark-up language (AIML) is an XML-based mark-up language that used for brain or knowledge of the chatbot which used the first time in ALICE bot [13, 14]. In other researches, using AIML for the brain of the chatbot is a success in creating interaction between bot and user.

The Nazief & Adriani algorithm is a stemming algorithm commonly used for Bahasa Indonesia, which is considered as the qualify one, compared to other stemming algorithms [15]. That algorithm used because chatbot using Bahasa Indonesia and Nazief & Adriani algorithm have higher accuracy for the Indonesian language stemmer than other algorithms such as porter [16]. Other works have also shown the potential of Nazief & Adriani algorithm in stemming natural languages in Bahasa Indonesia [17–19]. Furthermore, chatbot for related works of literature concerning the development chatbot in Bahasa Indonesia has also shown the use of chatbot in various fields [20–22]; however, the use of chatbot and Nazief & Adriani algorithm in the requirements engineering field is still rare to be found.

This research aims to bridge the gap between the stakeholders and the software engineers in terms of requirements gathering by developing a chatbot for the stakeholders to chat with when submitting preliminary requirements and/or user feedbacks regarding a software product. The chatbot uses AIML as its knowledge base, Nazief & Adriani stemming algorithm, and follows the User Stories format which will require the users to provide information regarding his/her role, needs, and business value in a natural conversation. The bot's performance was measured using several chatbot metrics, and a preliminary user acceptance test following the technology acceptance model was conducted for evaluation.

2. RESEARCH METHOD

In this research, the chatbot application built using the Nazief & Adriani stemming algorithm for the stemming process and AIML is used for knowledge of the chatbot. Chatbot used for gathering software requirements using user stories.

2.1. Chatbot design

In building the Android application with the chatbot included, the object-oriented programming approach is used. UML diagrammatic notations are used to design the application. Figure 1 shows the use case diagram of the chatbot. There are two kinds of users, the generic user, and the software developer. The generic users could perform two main activities which are to view the list of chat rooms in his/her application, and also to give user stories by communicating with the chatbot. The developers could do anything the generic user can do, and also, he/she could also view the submitted user stories as the requirements gathered for the corresponding application.

2.2. AIML

Our research uses AIML (Artificial Intelligence Markup Language) used for the knowledge (brain) of the chatbot. In AIML, for one tag category usually, it consists of tag pattern and tag template. Tag pattern is used for matching user input and tag template used for output from the bot. Figure 2 shows an example of the AIML of the chatbot application. That example shows the knowledge of the chatbot for users to input their role. After the user inputs their role, the submitted data will be caught by the wildcard symbol in tag pattern and will be set in variable temp. After that, in order to get a response from the chatbot, the value of the variable temp will be checked.

2.3. Text pre-processing

Text preprocessing process is done before parsing the message to the brain file. Firstly, normalization is done, which message will be converted to lower case. After normalizing the message, tokenizing will be done. Tokenizing is a process that split the message into tokens. Each token will be checked if there is a mistyped word. Process stop word removal will be done after checking the typo word after that stemming will be done with Nazief & Adriani stemming algorithm. Stemming starts by checking the combination prefix-suffix process. If a common combination detected, the suffix is then checked and stemmed, and after that, the prefix will be checked and stemmed, and finally, a new word will be received. If a unique combination is detected, the prefix will be checked, and stem and after that suffix will be checked

and stemmed, then again, a new word will be received. Figure 3 shows the activity diagram for the preprocessing process.

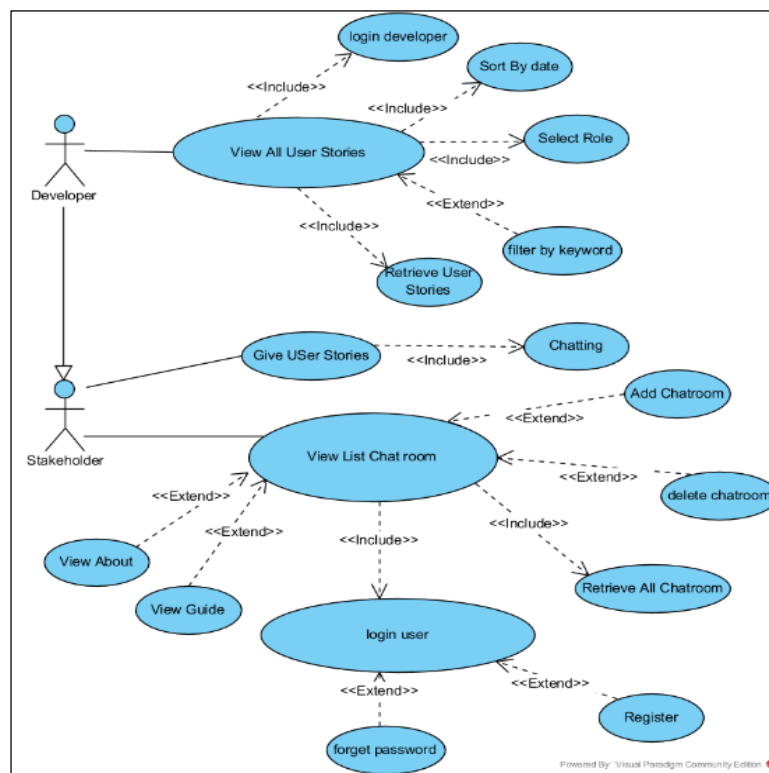


Figure 1. Use case diagram

```

<category>
  <pattern> saya * </pattern>
  <template>
    <think>
      <set name="temp"><star></set>
    </think>
    <condition name="temp">
      <li value = "dosen">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">dosen</set></think>
        Sebagai dosen, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li value = "guru">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">dosen</set></think>
        Sebagai dosen, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li value = "pengajar">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">dosen</set></think>
        Sebagai dosen, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li value = "mahasiswa">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">mahasiswa</set></think>
        Sebagai mahasiswa, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li value = "staff">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">staff</set></think>
        Sebagai staff, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li value = "baak">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">staff</set></think>
        Sebagai staff, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li value = "bauk">
        <think><set name="topic">start</set><set name="tempS">job</set><set name="job">staff</set></think>
        Sebagai staff, apa yang anda butuhkan/harapkan dari <get name="apps"/>? |<get name="job"/>
      </li>
      <li>
        <random>
          <li>Maaf. Profesi anda apa?
        </li>
        </random>
      </li>
    </condition>
  </template>
</category>
  
```

Figure 2. Example of AIML

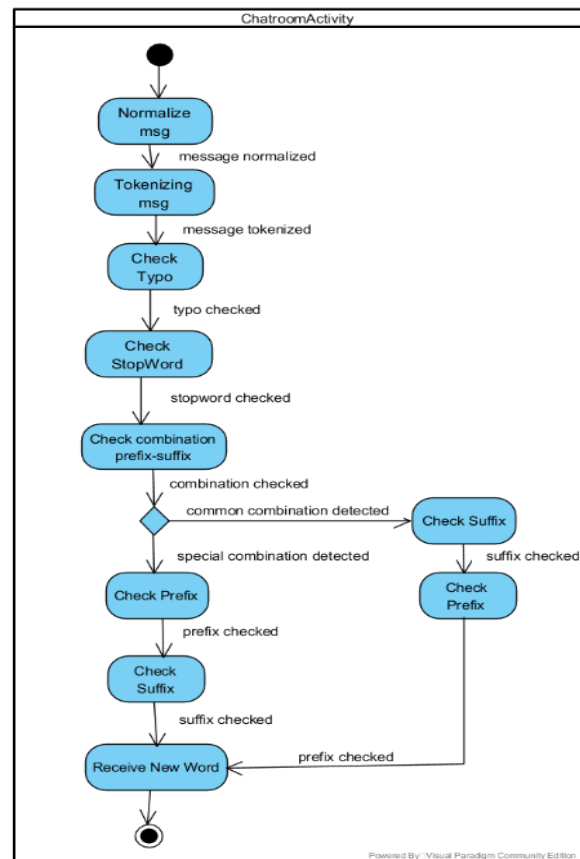


Figure 3. Activity diagram preprocessing process

2.4. User story

The user story concept is a part of the agile development method that was first proposed in extreme programming for gathering user requirements that are needed by the system [4]. It then becomes key in the software development process [23]. User stories focus on the interaction between the user or customer [3], and they will describe functions that are needed for the system in the form of a story [4]. Example form of the user story can be seen in Figure 4 [24]:

- a. <role> represents the role of the person that provide the story.
- b. <activity> represents the requirement.
- c. <business value> represents the reason or value of the above requirement.

As a <role> I can <activity> so that <business value>

Figure 4. User story

3. RESULTS AND ANALYSIS

The result of the research is a chatbot application that used for gathering software requirements using user stories and Nazief & Adriani stemming algorithm. In this chatbot AIML used for the chatbot knowledge. A user acceptance test is applied in this research to evaluate the chatbot's performance, perceived by the users.

3.1. Chatbot implementation

Figure 5 (a) display the splash screen of the chatbot applications. Splash screen will be show when user open the applications. Figure 5 (b) shows the chat room list menu. This menu show list chat room that user created. In this menu user can create or delete chat room. Chat room is represented by a card. Each card of display index of card, the name of the application and role of user.

Figure 6 shows the user story menu. The user story menu is for the developer so they can see a list of user stories that have been obtained from the user. Each user story represented by a card that every card shows the story, reason, application name, and the date that the user story obtained. In each card also have a delete button so the developer can delete the user story. At the top of the user story, there is a field that can be used by the developer to search the user story by application name, story, and reason. This list of the user story is already sorted by descending so the latest data will always be displayed in the first row, Figure 7 shows the chat rooms and an example of giving a user story. In this chat room, users can chat with the bot and give the user story. The first-time bot will ask the role of the user. After the user tells their bot, the role and the bot will then store the role information, and the user can start to give the user story. Every time the user gives the user the reason the bot will ask confirmation whether or not the user has given by the user correctly. If the story and the reason already correct, the user story will be accepted and saved.

3.2. Chatbot metrics

Several metrics were used to evaluate the bot's performance quantitatively. The metrics used include total elapsed time, total number of user turn, total number system turn, the total number of turns per task, dan total elapsed time per turn. Total elapsed time calculates the total elapsed time from when the user starts the conversation until when the user finishes submit a user story. The total number of user turn calculates the number of user chats from when the user starts the conversation until when the user finishes submit a user story. The total number of system's turn calculates the number of bot responses from when the user starts the conversation until when the user finishes submit a user story. Total elapsed time per turn calculates the elapsed time for a user to finish his/her turn; this is calculated by dividing the total elapsed time by the total number of turns. The metrics evaluate the provision of 30 user stories given by three users in which the time is measured manually using a timer. Table 1 shows the sample metric of a user when testing the chatbot application, performing a total of 8 tasks. Each task in Table 1 represents an activity in which a user is communicating using the chatbot to provide user stories, with the following numbers are calculated for the metrics.



Figure 5. (a) Splashscreen (b) Chat room list menu

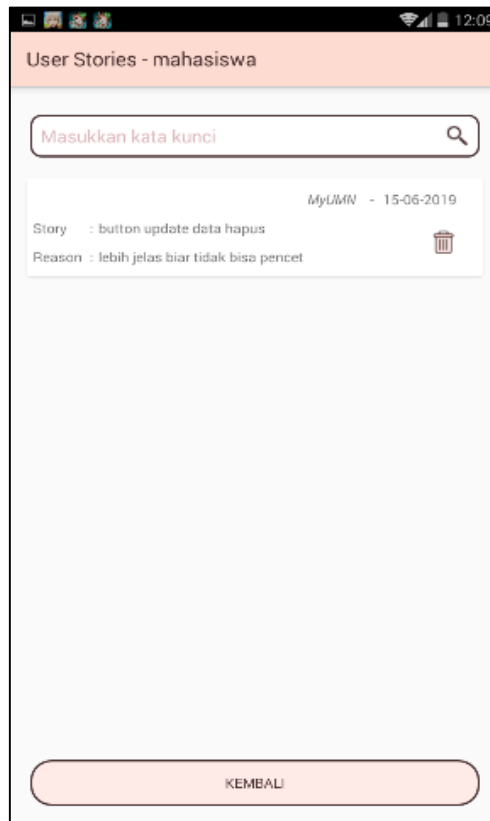


Figure 6. List of user stories

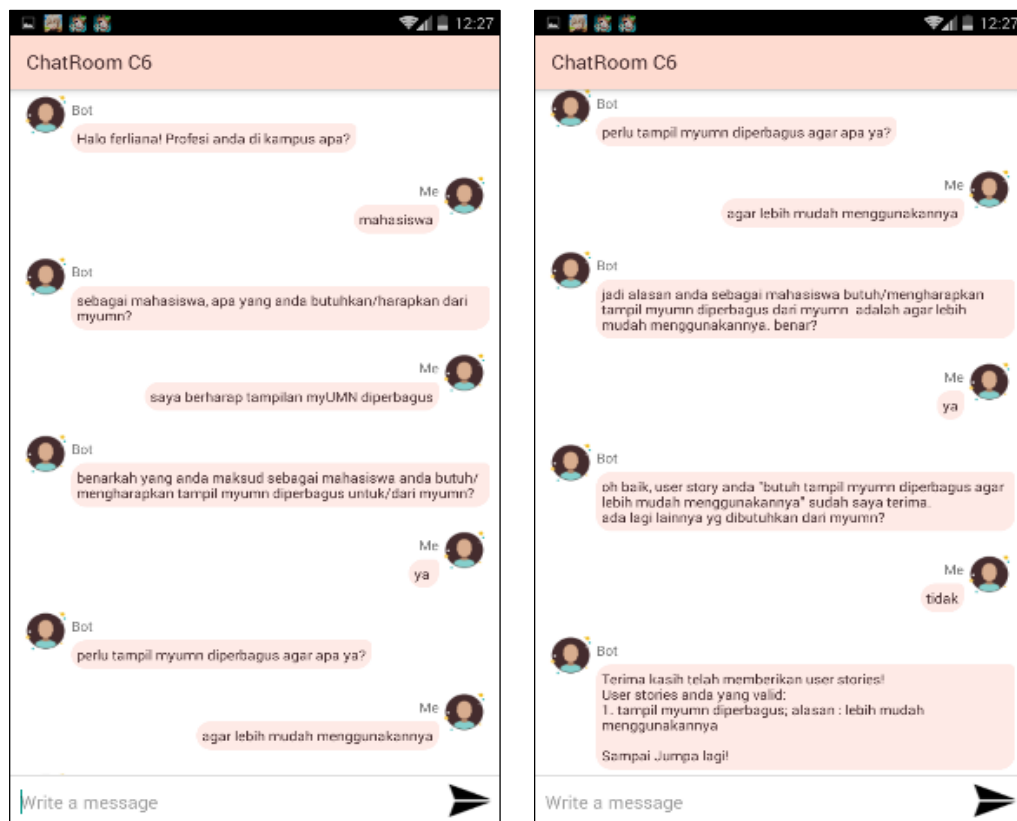


Figure 7. Chat room

Table 1. Chatbot metrics

	Total elapsed time (seconds)	Total number of user turn	Total number of system turn	Total turns per task	Total elapsed time per turn (seconds)
Task 1	50.70	6	6	12	4.23
Task 2	37.18	7	7	14	2.66
Task 3	36.14	6	6	12	3.01
Task 4	32.00	6	6	12	2.67
Task 5	39.24	6	6	12	3.27
Task 6	44.57	6	6	12	3.71
Task 7	44.44	7	7	14	3.17
Task 8	48.68	6	6	12	4.06
Total	332.95	50	50	100	26.77
Average	41.62	6	6	12	3.35

3.3. Preliminary user acceptance test

A preliminary user acceptance test was conducted to evaluate the perceived usefulness (PU), user attitude towards (A), behavioral intention to use (BI), and perceived ease of use (PEOU), following the technology acceptance model proposed by Davis [25]. The questionnaire uses a five-point Likert scale used in order to determine the level of user acceptance, which is comprised of strongly agree (5), agree (4), neutral (3), disagree (2), and strongly disagree (1). Thirty-three respondents participated in this questionnaire. In order to test the capabilities of the chatbot, we use a university information system, as the case study for which the users will provide feedback.

User acceptance test is done to 33 users using a web-based university information system, called MyUMN, as a test case for which the users will provide user stories on. The respondent is MyUMN's users which are student, lecturer or university staffs. The respondent's age is above 20 years old, both male and female respondents. Firstly, respondents were asked to download the APK file for the chatbot application. Then, users experimented with the app, based on the application description explaining about user story and what the chatbot aims to perform. After the users tried and experimented with giving feedbacks regarding the MyUMN web application and communicating with the chatbot, questionnaires were distributed, and interviews were conducted to get feedback from the users, mainly focusing on the four aspects of the technology acceptance model mentioned previously. The questionnaires' answer sheet is formed based on the intervals using the Likert Scale, with interpretation, as shown in Table 2 below.

Based on the questionnaires filled by the users after using the chatbot application, we calculated the average score from the questionnaire for each factor in the technology acceptance model. Results are as follow, 83.79% for user attitude towards the chatbot application, 85.45% for perceived usefulness, 84.55% for perceived ease of use, and 83.03% for users' behavioral intention to use. The graphical result of user acceptance test can be seen in Figure 8. Interviews also shows that the users' as stakeholders of the MyUMN web application. The overall results seems not to differ from one aspect to another, however if we look closely, perceived usefulness achieved the highest score, this aspect measures the users' acceptance regarding how well the chatbot is perceived to be useful to support the requirements gathering activity (from the developers point of view) and the feedback provision activity (from the users' perspective) by providing a possibility of a natural communication to structure the feedbacks following the user story format, eliciting the feedback providers' role, their needs, and the business value regarding the needs/requirements provided.

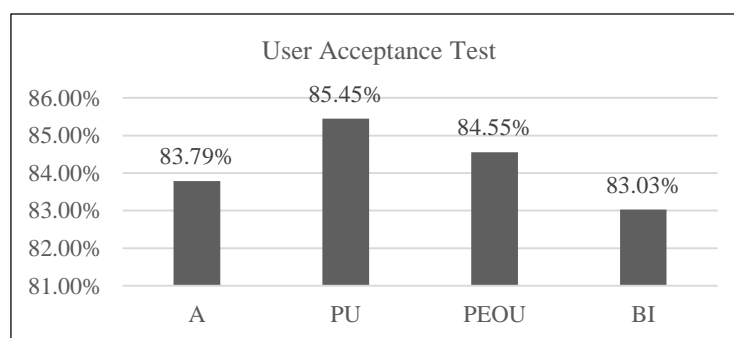


Figure 8. User acceptance result

The behavioral intention to use scored the lowest of the four aspects, however, looking at the interpretation score shown in Table 2, it still managed to be able to be interpreted as Strongly Agree. Interviews conducted after the questionnaires-filling activity shows the users' motivation and intention to use the chatbot again in the future.

Table 2. User acceptance test score interpretation

Interpretation	Percentage Score	Value
Strongly Disagree	0%-19.99%	1
Disagree	20%-39.99%	2
Neutral	40%-59.99%	3
Agree	60%-79.99%	4
Strongly Agree	80%-100%	5

4. CONCLUSION

The conducted research has developed a chatbot application based on the Android mobile platform, to support the requirements gathering activity by providing ways to bridge the communication gap between the developers of a software product and their stakeholders. The requirements that are gathered are then naturally enforced by the chatbot to follow the user story format containing the users' role, requirements/needs, and its business value. The chatbot also implemented the Nazief & Adriani stemming algorithm for the specific need of stemming words written in Bahasa Indonesia. Furthermore, it utilizes the Artificial Intelligence Markup Language for the bot's knowledge base in which the bot response is decided, given a user chat has been submitted. A preliminary user acceptance test based on the technology acceptance model shows promising results. The test focused on evaluating four aspects of user acceptance which are the user attitude toward the chatbot application, perceived usefulness, perceived ease of use, and the users' behavioral intention to use. The chatbot received scores for the four aspects as follow, 83.79% for user attitude towards the chatbot application, 85.45% for perceived usefulness, 84.55% for perceived ease of use, and 83.03% for users' behavioral intention to use. Future works include employing a more advanced natural language processing method for feedbacks written in Bahasa Indonesia, developing the web dashboard for the superuser (developers) to manage the AIML brain file and manage submitted feedback by filtering and classifying them into meaningful information for the software engineers. Furthermore, a more thorough user acceptance test could also be conducted to further evaluate the chatbot's performance and its users' perspective toward the chatbot.

ACKNOWLEDGEMENT

This research was supported by the Mobile Development Laboratory in Universitas Multimedia Nusantara. We also thank our colleagues from the Faculty of Engineering and Informatics who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

REFERENCES

- [1] Pressman RS, "Software Engineering A Practitioner's Approach 7th Edition," *Software Engineering A Practitioner's Approach 7th Ed-Roger S. Pressman*. 2010.
- [2] Sommerville I, "Software Engineering," *Software Engineering*. 2010.
- [3] Lamsweerde A Van, "Requirements Engineering: From System Goals to UML Models to Software Specifications," *Change*, 2010.
- [4] O'hEocha C, Conboy K, "The role of the user story agile practice in innovation," *Lecture Notes in Business Information Processing*, pp. 20-30, 2010.
- [5] Abdul-Kader SA, John D, "Survey on Chatbot Design Techniques in Speech Conversation Systems," *Int J Adv Comput Sci Appl*. 2015.
- [6] Huang J, Zhou M, Yang D, "Extracting chatbot knowledge from online discussion forums," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 6-12, 2007.
- [7] Shawar B, Atwell E, "Using dialogue corpora to train a chatbot," *Proc Corpus Linguist*, pp. 681-690, 2003.
- [8] Oracle, "Can Virtual Experiences Replace Reality? The future role for humans in delivering customer experience," *Oracle*, pp. 1-19, 2016.
- [9] Nguyen M-H, "From Fortune 500s to small businesses, real businesses are already using chatbots to improve their service," 2017.
- [10] Beaver L, "Chatbots are gaining traction Internet," 2017. [Online]. Available: <https://www.businessinsider.com/chatbots-are-gaining-traction-2017-5/?IR=T>. Accessed: 2 October 2018.

- [11] Van Eeuwen M, "Mobile conversational commerce: messenger chatbots as the next interface between businesses and consumers," *Univ Twente*. 2017.
- [12] Databoks.co.id, "Pengguna Ponsel Indonesia Mencapai 142% dari Populasi [Internet]," 2017. [Online]. Available: <https://databoks.katadata.co.id/datapublish/2017/08/29/pengguna-ponsel-indonesia-mencapai-142-dari-populasi>. Accessed: 2 October 2018.
- [13] Marietto B, Aguiar M das G, Aguias RV, "Artificial Intelligence Markup Language: A Brief Tutorial," *Int J Comput Sci Eng Surv*. 2013.
- [14] Wallace R, "The elements of AIML style," *Alice AI Found*. 2003.
- [15] Mardiana T, Adji TB, Hidayah I, "Stemming influence on similarity detection of abstract written in Indonesia," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 14, no. 1, pp. 219-227, 2016.
- [16] Agusta L, "Comparison of Porter's Stemming Algorithm with Nazief & Adriani's Algorithm for Stemming Indonesian Text Documents (in Bahasa: Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia)," *Konf Nas Sist dan Inform*, 2009.
- [17] Hidayatullah AF, Ratnasari CI, Wisnugroho S, "Analysis of Stemming Influence on Indonesian Tweet Classification," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 14, no. 2, pp. 665-673, 2016.
- [18] Watequlis Syaifudin Y, Saputra PY, Puspitasari D, "The implementation of web service based text preprocessing to measure Indonesian student thesis similarity level," *MATEC Web of Conferences*, 2018.
- [19] Aulia A, Khairani D, Hakiem N, "Development of a retrieval system for Al Hadith in Bahasa (case study: Hadith Bukhari)," *2017 5th International Conference on Cyber and IT Service Management, CITSM 2017*.
- [20] Purbasari, I. Y., Anggraeny, F. T., Maharani, M. F., "Replies Identification in Question Answering System using Vector Space Model," *Atlantis Highlights in Engineering (AHE)*, vol. 1, pp. 459-464, 2018.
- [21] Toxtli C, Monroy-Hernández A, Cranshaw J, "Understanding chatbot-mediated task management," *Conference on Human Factors in Computing Systems – Proceedings*, 2018.
- [22] Tedjoprano, M. L., Wijaya, A., Santoso, L. H., Suhartono, D, "Correcting Typographical Error and Understanding User Intention in Chatbot by Combining N-Gram and Machine Learning Using Schema Matching Technique," *International Journal of Machine Learning and Computing*, vol. 9, no. 4, pp. 471-476, 2019.
- [23] Rees M. J., "A feasible user story tool for agile software development?," *Proceedings-Asia-Pacific Software Engineering Conference, APSEC*, 2002.
- [24] Zeaaraoui A., Bougroun Z., Belkasm M. G., Bouchentouf T., "User stories template for object-oriented applications," In: *2013 3rd International Conference on Innovative Computing Technology, INTECH 2013*.
- [25] Davis F. D., Bagozzi R. P., Warshaw P. R., "User Acceptance of Computer Technology: A Comparison of Two Theoretical Models," *Manage Sci*, 1989.

BIOGRAPHIES OF AUTHORS



Ferliana Dwitama is graduated from Universitas Multimedia Nusantara in November 2019 with a bachelor's of Computer Science. She has background in information technology and holds keen interests in the area of requirements engineering and chatbot.



Andre Rusli received his Master's of Science degree in Information Environment from Tokyo Denki University, Japan, in 2017, focusing in the Software Requirements Engineering field. He is currently a lecturer and researcher in Universitas Multimedia Nusantara and also serving as the head coordinator of the Mobile Development Laboratory. His research interests include requirements engineering in software application development, natural language processing, and human computer interaction.